OCR
Oxford Cambridge and RSA
An OCR endorsed teaching and learning tool

# OCR A Level Computer Science

H446 – Paper 1

# Number systems, ASCII and Unicode

Unit 6
Data types

6

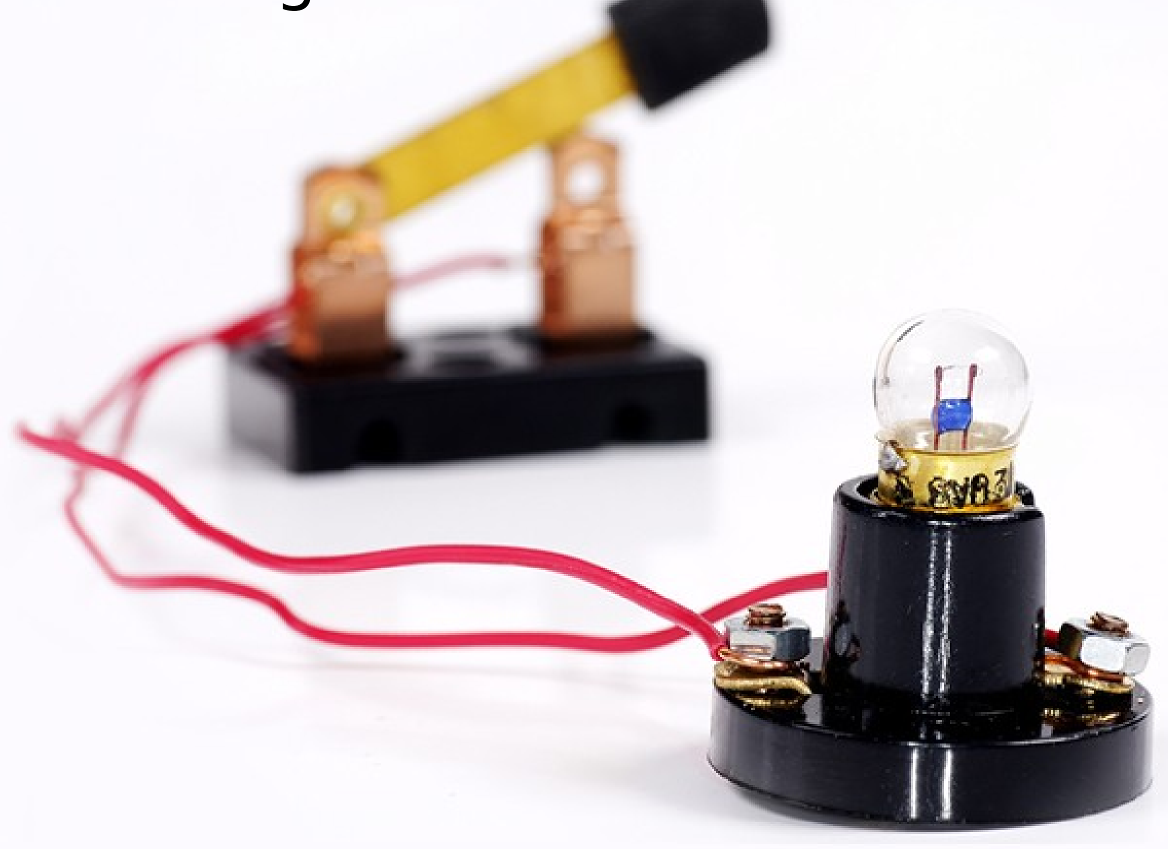
PG ONLINE

# Objectives

- Define a bit as a 1 or a 0, and a byte as a group of eight bits
- Know that $2^n$ different values can be represented with *n* bits
- Use names, symbols and corresponding powers of 2 for binary prefixes e.g. Ki, Mi
- Differentiate between the character code of a decimal digit and its pure binary representation
- Describe how character sets (ASCII and Unicode) are used to represent text

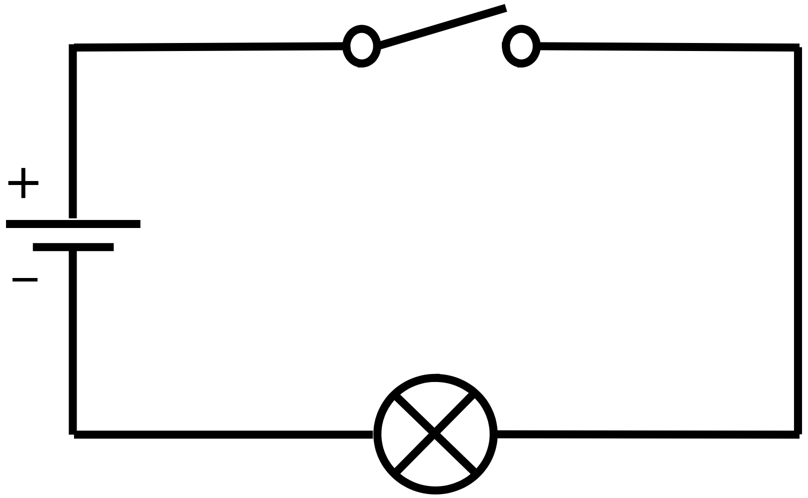
# Creating a circuit

- Computers comprise billions of switches to turn voltage on and off



# Circuits

- Use the keywords below to explain how this electrical circuit works:



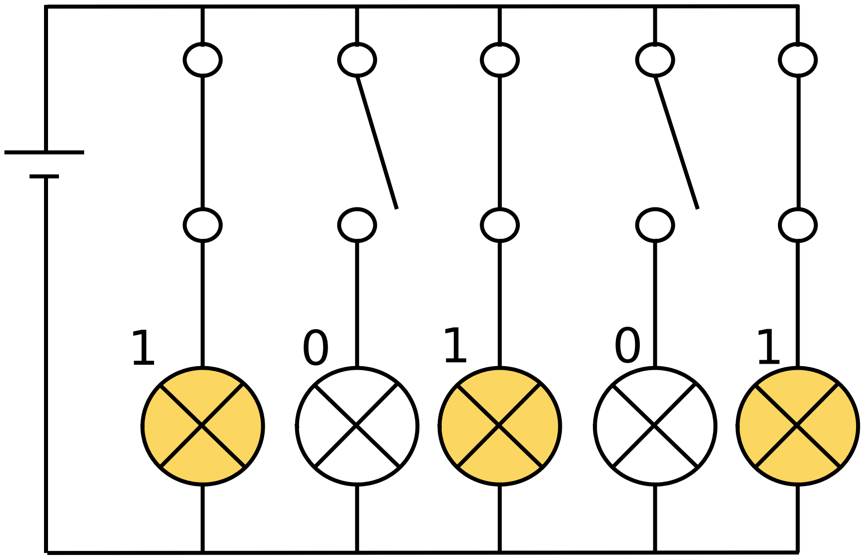

**Battery** **Switch** **Lamp**

# Electricity

- A computer is an electrical device that works in the same way as a light bulb connected to a battery
  - Data is stored and processed using combinations of ON and OFF voltages, equivalent to a lamp turning on and off where **ON = 1** and **OFF = 0**
  - These voltages are 'transferred' around the parts of the computer using wires

# Representing values

- Consider the value $21_{10}$
  - In binary we would represent this value as $10101_2$
  - As an electrical circuit this could be represented as:



# Binary digits

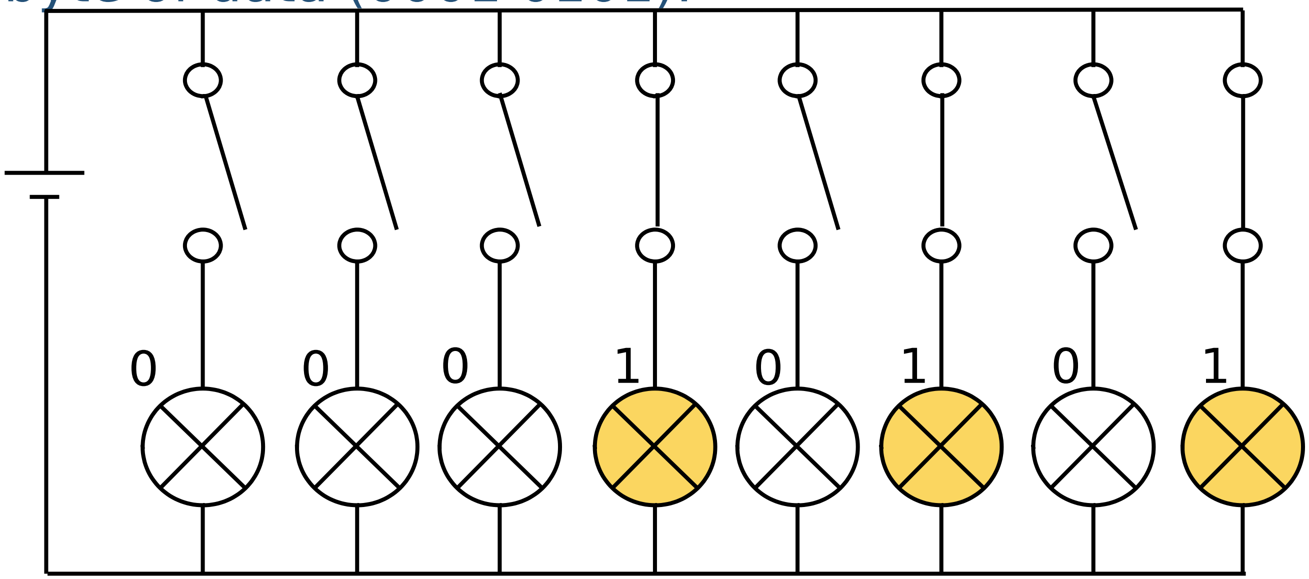
- Each individual digit in a binary value is referred to as a **bit**, (from the term **b**inary dig**it**)
- In a computer we can represent binary values by using ON and OFF voltage signals for each individual bit
  - For *n* bits a computer can produce $2^n$ different combinations of values
  - How many combinations are there using 3 bits and what are they?

# Bytes

- As a binary value gets larger, more bits are needed to store the number
- A computer has fixed wiring that cannot be adjusted to accommodate more bits; instead it works with bits grouped together into units called **bytes**
- A **byte** is a collection of 8 bits
- Two or more bytes can be grouped together to hold larger values

# Fixed bits

- Consider the previous circuit model for $21_{10}$
  - By padding the data with three leading 0s, its definition can be extended to describe a complete byte of data (0001 0101):



# Large values

- Computers process and store large amounts of bytes, often in the order millions or billions
  - When dealing with large quantities it is more convenient to summarise this using number **prefixes**
  - A common example of this is the kilogram **k** which is equivalent to 1000 grams:

# Prefixes for bytes

- The same number prefixes for decimal values can be used to summarise large quantities of bytes
- Common prefixes include:

| Prefix | Symbol applied to Bytes | Multiple |
|---|---|---|
| kilo | kB | $10^3$ = 1,000 |
| meg a | MB | $10^6$ = 1,000,000 |
| giga | GB | $10^9$ = 1,000,000,000 |
| tera | TB | $10^{12}$ = 1,000,000,000,000 |

# Incorrect prefixes

- Traditionally computer scientists used these same number prefixes to refer to groups of bytes
- These are not the same as their decimal equivalents
    - Base 2 was used as the multiplier instead of base 10
    - For example, a kilobyte was used as a representation of $2^{10}$ bytes, megabyte $2^{20}$ and so on
    - So, 1 kB was equivalent to 1024 Bytes and 1MB to 1,048,576 Bytes or 1024 kB

# New prefixes

- To eliminate this confusion, in 1998 the International Electrotechnical Commission (IEC) established different prefixes to represent multiples of base 2:

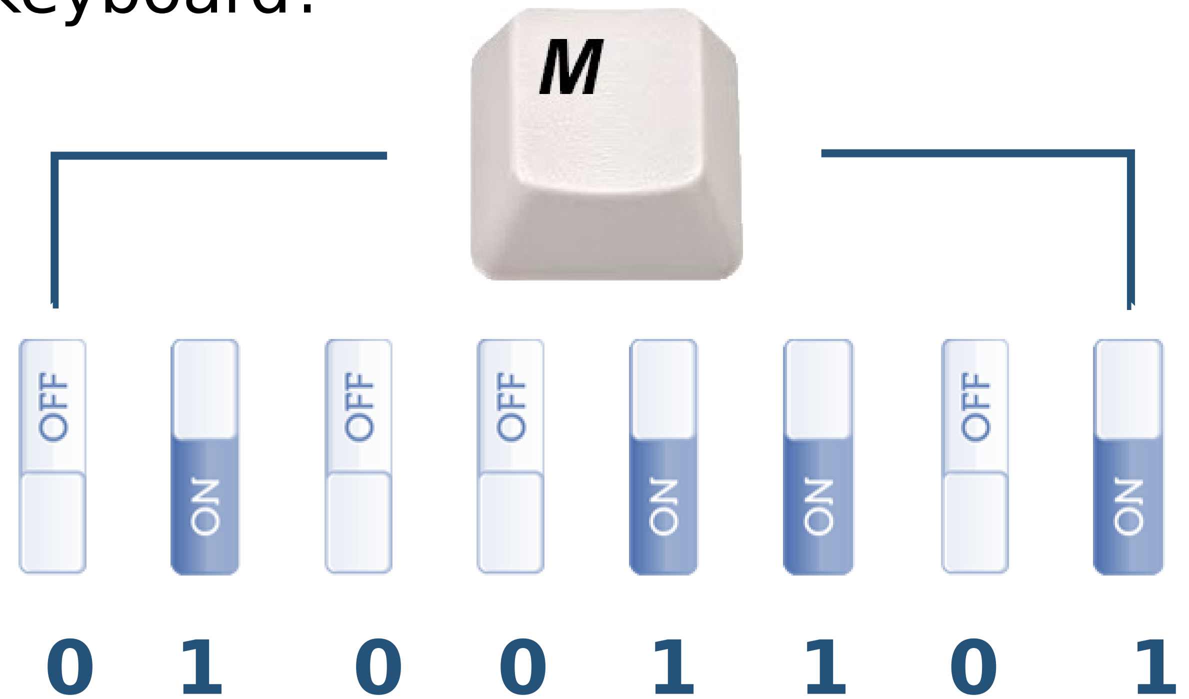
| Prefix | Symbol applied to Bytes | Multiple |
|---|---|---|
| kibi | KiB | $2^{10} = 1{,}024$ |
| mebi | MiB | $2^{20} = 1{,}048{,}576$ |
| gibi | GiB | $2^{30} = 1{,}073{,}741{,}824$ |
| tebi | TiB | $2^{40} = 1{,}099{,}511{,}627{,}776$ |

# Activity

- Complete **Worksheet 2**, **Task 1**

# Representing text characters

- If a computer understands only 1s and 0s, what happens when the ‘M’ key is pressed on the keyboard?



# ASCII Code

- In 1963 the **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange, (**ASCII**,) was established to encode symbols found in the English alphabet
  - It was composed of a 7 bit character set giving $2^7$ (128) possible binary codes
- What are the limitations of using a 7 bit

| Decimal | Binary | Character | Decimal | Binary | Character | Decimal | Binary | Character |
|---|---|---|---|---|---|---|---|---|
| 32 | 00100000 | space | 64 | 01000000 | @ | 96 | 01100000 | ' |
| 33 | 00100001 | ! | 65 | 01000001 | A | 97 | 01100001 | a |
| 34 | 00100010 | " | 66 | 01000010 | B | 98 | 01100010 | b |
| 35 | 00100011 | £ | 67 | 01000011 | C | 99 | 01100011 | c |
| 36 | 00100100 | $ | 68 | 01000100 | D | 100 | 01100100 | d |
| 37 | 00100101 | % | 69 | 01000101 | E | 101 | 01100101 | e |
| 38 | 00100110 | & | 70 | 01000110 | F | 102 | 01100110 | f |

# Representing characters

- Every character on the keyboard is represented by a binary value
- Uppercase letters (capitals) have different values from lowercase characters
- Punctuation symbols have their own characters
- How many characters are there on a standard keyboard? How many bits would be required to represent this many combinations?
- What character is represented by 0100000 (32)?

# ASCII Table

- Why does a space need a character? (Code 32)
- What happens if you press ***ALT+65*** on a keyboard?
- The first 32 codes are control characters, for example:
    - Backspace (Code 8)
    - Carriage Return or Enter (Code 13)
    - Escape (Code 27)
- An eighth bit was later introduced for extra characters such as ©, ® etc.

# Character form of decimal digits

- Numeric characters are also encoded
  - The code 0111001 represents the character ‘9’ in ASCII
  - The binary byte representing ‘9’ would be $00001001_2$
- What are the implications of this difference?
- What will this code output?

```
print ('9 + 3 =',9 + 3)
print ('"9" + "3" = ',"9" + "3")
print ('ord("9") + ord("3") = ',ord("9") +
ord("3"))
```

# Unicode

- The Unicode system was introduced to standardise the encoding of characters from all languages
  - Unicode can apply a variable length encoding of either 16 bits or 32 bits
  - In order to improve adoption of this new standard the first 128 Unicode characters were set to be the same as the ASCII character set
- What is the disadvantage of using up to 4 bytes per character?

# Advantage of Unicode

- In Unicode, every character in every language in the world, every mathematical and scientific symbol, etc. can be represented

  Español

  □□□

  Македонски

  □□□□□□ □□

  ελληνικά

# Worksheet 2

- Complete **Task 2**

# Plenary

- Electrical signals operated by switches ‘create’ a binary pattern
- New prefixes e.g. Kibi and Mebi describe $2^n$ bytes
- ASCII and Unicode are used to represent characters
- An ASCII digit is not the same as its direct binary translation
- Unicode provides a unique way of encoding the alphabet and characters of every nation in the world

PG ONLINE